SOZLab

Linux GUI + CLI for solvent occupancy zone analysis

Version: 0.1.1 | Date: January 23, 2026

Table of Contents

SOZLab Tutorial 2

What SOZLab does . . . 2

Quick Start (CLI + GUI) . . . 3

Using your own files (GROMACS-focused) . . . 4

PBC preprocessing (why it matters) . . . 5

Seeds and Selection Language . . . 5

SOZ Logic Trees (Wizard + Advanced) . . . 6

Results and outputs . . . 7

Plots and visualizations . . . 9

Frame extraction (GUI + CLI) . . . 10

Reproducibility and provenance . . . 11

Scientific QC / Reliability . . . 11

Troubleshooting . . . 12

SOZLab Tutorial

What SOZLab does

SOZLab is a Linux-only GUI + CLI for solvent occupancy zone (SOZ) analysis of MD trajectories. It evaluates, per frame, which solvent residues satisfy a user-defined logic tree around one or more seed selections. The outputs include per-frame occupancy counts, entry and exit events, per-solvent residence summaries, and optional plots, reports, and extracted trajectories.

Operational definition of a SOZ

A SOZ is the set of solvent residues that satisfy a logic tree in a given frame. Nodes in that tree are built from seed selections (selection A and optional selection B) and distance or shell criteria. The evaluated set is a set of solvent resindices, not atoms.

Per-frame metrics (definitions)

1 **frame**: zero-based frame index from the trajectory. This is the actual trajectory frame index after applying **analysis.frame_start**, **analysis.frame_stop**, and **analysis.stride**.

1. **time**: **ts.time** from MDAnalysis for that frame. The raw values are written to CSV as-is (typically ps for GROMACS trajectories).
1. **n_solvent**: count of solvent residues in the SOZ for that frame. This is an integer count per frame.
1. **entries**: count of solvent residues that appear compared to the previous sampled frame.
1. **exits**: count of solvent residues that disappear compared to the previous sampled frame.

Residence summaries (**per_solvent.csv** and **summary.json**) report continuous and intermittent residence segments. Continuous segments end as soon as a solvent leaves. Intermittent segments allow gaps of up to **analysis.gap_tolerance** frames before breaking the residence.

Time basis (frame index vs time)

1. SOZLab uses **ts.time** from MDAnalysis for the **time** column and for time-based statistics. The **summary.json** includes **time_unit** from **universe.trajectory.units** when available, otherwise it falls back to **ps**.
1. The GUI timeline and plots convert **time** to nanoseconds by dividing by 1000 and label the x-axis as **ns**. If your trajectory time units are not ps, confirm the time basis before interpreting plots.

Length units (Angstrom vs nm)

1. User inputs for cutoffs accept **unit: "A"** (angstrom) or **unit: "nm"** in the project file.
1. SOZLab converts user units to internal angstrom values before distance calculations.
1. GROMACS conventions use nm in input files; SOZLab will convert nm to angstrom internally. Conversion: **1 nm = 10 A**.

Quick Start (CLI + GUI)

CLI quick start (sample project)

Assumes the environment is installed and **sozlab-cli** is on PATH.

```
sozlab-cli run --project examples/sample_project.json --output out --progress --report
```

Expected outputs in **out/**:

1. **metadata.json**
1. **soz_shell_only/** and **soz_shell_and_selection_b/**
1. **report/report.md** (and report figures)

Validation (optional sanity check):

```
sozlab-cli validate --project examples/sample_project.json --max-frames 50
```

Extraction (example):

```
sozlab-cli extract --project examples/sample_project.json --soz shell_only --rule
```

```
"n_solvent>=1" --min-run 5 --gap 1 --out out/extracted
```

GUI quick start (sample project)

1) Launch: **sozlab-gui**. 2) Click **Load** and open **examples/sample_project.json**. 3) In the left panel, confirm **Frame start/stop/stride** and **Output Settings > Output directory**. 4) Click **Run** (header bar). Status appears in the bottom status bar. 5) Review tabs: **Overview**, **QC Summary**, **Timeline**, **Plots**, **Tables**, **Report**, **Logs**, **Extract**. 6) Use **Run Project Doctor** in the left panel to re-check selections and solvent settings.

Output directory precedence

1. **project.outputs.output_dir** is the default base directory for logs and analysis outputs.
1. **sozlab-cli run --output OUT** overrides **outputs.output_dir** for that run. Logs and all outputs are written under **OUT/**.
1. **sozlab-cli extract --output OUT** only overrides the log directory (where **sozlab.log** is written).
1. **sozlab-cli extract --out OUT_EXTRACT** controls the extraction output directory (frames, reference PDB, manifest).
1. The GUI **Output Settings > Output directory** controls the same output base for runs and exports.

Using your own files (GROMACS-focused)

Required file combinations

The project file must provide a topology and, optionally, a trajectory.

```
Topology   | Trajectory | Notes
---------- | ---------- | ------------------------------------------------------------
.tpr       | .xtc       | Preferred for reproducibility; topology and trajectory match.
.gro       | .xtc       | Common when .tpr is missing; MDAnalysis infers topology.
.pdb       | .xtc       | Lightweight tests; ensure atom order matches the trajectory.
.pdb       | (none)     | Single-frame checks only; no time series or entry/exit trends.
```

Pre-flight verification checklist

1. Atom count matches: topology and trajectory must report the same number of atoms.
1. Solvent and ion resnames exist: **solvent.water_resnames** (and **ion_resnames** if **include_ions=true**) must match residues in the topology.
1. Seed selections resolve to expected counts: use **require_unique** for single atoms and check **expect_count** when needed.
1. Box vectors are present: QC warning **No valid box vectors found** indicates unreliable PBC distances.
1. Units are valid: cutoffs must use **A** or **nm** only.

Integrity checks (MDAnalysis + optional GROMACS)

Minimal MDAnalysis check:

```
import MDAnalysis as mda
u = mda.Universe("/path/to/topol.tpr", "/path/to/traj.xtc")
print("atoms", len(u.atoms), "frames", len(u.trajectory))
print("resnames", sorted({str(res.resname) for res in u.residues})[:20])
print("selection count", len(u.select_atoms("protein and resid 10 and name CA")))
```

Optional GROMACS check (only if **gmx** is available):

```
gmx check -f traj.xtc
```

If supported in your GROMACS build, **gmx report-methods** can confirm that **.tpr** and **.xtc** were generated consistently.

PBC preprocessing (why it matters)

SOZLab relies on consistent solvent coordinates. Discontinuities from periodic boundary jumps can corrupt entry/exit counts and distance-based selection.

Recommended recipes:

```
# Remove jumps so molecules are continuous in time
# Useful when molecules jump across the box but you want continuous trajectories.
gmx trjconv -s topol.tpr -f traj.xtc -o traj_nojump.xtc -pbc nojump

# Keep molecules whole and wrapped in the primary box
# Useful before distance-based analyses in a compact box.
gmx trjconv -s topol.tpr -f traj.xtc -o traj_mol.xtc -pbc mol -center
```

Guidance:

1. Use **-pbc nojump** when you want continuous trajectories without wrapping molecules back into the primary box.
1. Use **-pbc mol** when you want intact molecules wrapped into the primary box for distance calculations.
1. Adding **-center** recenters a selection (usually protein) and changes the absolute coordinates for all other atoms.

Warning: PBC handling changes the geometry used for distance calculations. Always apply the same preprocessing to any analysis you compare.

Seeds and Selection Language

Selections are MDAnalysis selection strings. SOZLab also supports structured fields (**resid**, **resname**, **atomname**, **segid**, **chain**) which are converted to a selection string.

Curated selection examples

1) **protein and resid 10 and name CA** 2) **resname LIG and name O1** 3) **segid A and resid 220 and name NE2** 4) **chainID A and resname HEM and name FE** 5) **resname HIS and name NE2** 6) **resname HSD and name ND1** 7) **resid 45 and (name OE1 or name OE2)** 8) **resname SOL and name OW** 9) **index 100:120** 10) **bynum 3485** 11) **protein and around 5.0 (resname LIG)** 12) **resname HS* and name NE2**

Selection debugging playbook

1) Start broad: confirm a region with **protein and resid 1:10**. 2) Narrow with identifiers: add **segid** or **chainID** before enforcing **require_unique**. 3) Verify uniqueness: **require_unique** selections must resolve to exactly one atom. 4) Check resname variants: histidine tautomers often appear as HSD/HSE/HSP. 5) Use **Selection Tester** in the GUI to confirm counts and see suggestions.

SOZ Logic Trees (Wizard + Advanced)

Node types (one sentence each)

1. **shell**: returns solvent residues within one or more radial cutoffs around a selection; multi-shells are combined.
1. **distance**: returns solvent residues within a cutoff of a selection.
1. **and**: intersection of child node results.
1. **or**: union of child node results.
1. **not**: set difference between all solvent residues and the child node result.

Canonical shell-only example (JSON)

```
{
  "name": "shell_only",
  "description": "Oxygen shell around selection A",
  "root": {
    "type": "shell",
    "params": {
      "selection_label": "selection_a",
      "cutoffs": [3.5, 5.0],
      "unit": "A",
      "atom_mode": "O"
    },
    "children": []
  }
}
```

Shell AND selection B distance example (JSON)

```
{
  "name": "shell_and_selection_b",
  "description": "Shell around A that is also near selection B",
  "root": {
    "type": "and",
    "children": [
      {
        "type": "shell",
        "params": {
          "selection_label": "selection_a",
          "cutoffs": [3.5, 5.0],
          "unit": "A",
          "atom_mode": "O"
        },
        "children": []
      },
      {
        "type": "distance",
        "params": {
          "selection_label": "selection_b",
          "cutoff": 4.0,
          "unit": "A",
          "atom_mode": "O"
        },
        "children": []
      }
    ]
  }
}
```

Common mistakes and QC warnings

1. **Selection 'X' starts with an operator**: selection strings cannot start with **and**, **or**, or **not**.
1. **Selection 'X' resolved to 0 atoms**: the selection does not match the topology; use Selection Tester to inspect resnames, resids, segids.
1. **Selection 'X' expected unique atom but resolved N atoms**: add **segid** or **chainID**, or disable **require_unique**.
1. **Water resnames did not match any residues**: update **solvent.water_resnames** to match your topology.
1. **Water oxygen names did not match any atoms; O-mode will fall back to all atoms**: update **water_oxygen_names**.
1. **No valid box vectors found; PBC distances may be unreliable**: rewrap or use a trajectory with box dimensions.
1. **Unsupported unit 'X'**: use **A** or **nm** only.

Results and outputs

Output map

```
Path                                  | Meaning
------------------------------------- |
-------------------------------------------------------------
metadata.json                          | Project snapshot, warnings, and QC summary.
soz_<name>/per_frame.csv              | Per-frame time series (frame, time, n_solvent,
entries, exits).
soz_<name>/per_solvent.csv            | Per-solvent residence and occupancy summary.
soz_<name>/summary.json               | Aggregated SOZ summary metrics.
soz_<name>/min_distance_traces.csv    | Min distance traces per selection (if enabled).
bridge_<name>/per_frame.csv           | Bridge-specific per-frame table.
bridge_<name>/per_solvent.csv         | Bridge-specific per-solvent table.
hydration_<name>/hydration_table.csv  | Residue hydration table (if configured).
report/report.md                      | Report text (and figures).
report/report.html                    | Report HTML (only if report_format=html).
```

per_frame.csv columns

1. **frame**: trajectory frame index (after start/stop/stride).
1. **time**: raw MDAnalysis time for the frame.
1. **n_solvent**: integer count of solvent residues in the SOZ.
1. **solvent_ids**: semicolon-delimited solvent IDs; each ID is **resname:resid:segid**.
1. **entries**: number of solvent residues that appear relative to the previous sampled frame.
1. **exits**: number of solvent residues that disappear relative to the previous sampled frame.
1. Additional columns may appear when **analysis.store_min_distances=true** (one column per selection label with the minimum distance to any solvent residue in the SOZ for that frame).

Notes:

1. **entries** and **exits** are computed between sampled frames. If **analysis.stride > 1**, events can be missed and counts reflect changes between sampled frames.
1. Disabling per-frame output (**--no-per-frame**) removes this table, which also disables several GUI plots.

per_solvent.csv meaning

1. **solvent_id**: stable ID (**resname:resid:segid**).
1. **frames_present**: number of frames where the solvent is in the SOZ.
1. **occupancy_pct**: percent of sampled frames where the solvent is in the SOZ.
1. **entries**: number of entry events for that solvent.
1. **mean_res_time_cont**, **median_res_time_cont**: continuous residence times, in trajectory time units.
1. **mean_res_time_inter**, **median_res_time_inter**: intermittent residence times (gap-tolerant), in trajectory time units.

summary.json meaning

Key fields:

1. **n_frames**: number of sampled frames.
1. **frame_stride**: stride used for sampling.
1. **dt**: median time step between sampled frames.
1. **time_unit**: time unit string from MDAnalysis (falls back to **ps**).
1. **occupancy_fraction**: fraction of sampled frames with **n_solvent > 0**.
1. **mean_n_solvent**, **median_n_solvent**, **max_n_solvent**: occupancy summary statistics.
1. **n_solvent_hist**: mapping from **n_solvent** value to count of frames.
1. **entry_rate**, **exit_rate**: mean entry or exit events per sampled frame.

Plots and visualizations

Timeline (Occupancy)

1. X-axis: time in ns (GUI divides **time** by 1000 and labels ns).
1. Y-axis: selected metric (default is **n_solvent**).
1. Use **Mean line** and **Median line** toggles for reference; these add dashed lines for the selected metric.
1. Interpretation limits: a flat high **n_solvent** can be either stable occupancy or smeared events from a large stride.

Entry/Exit timeline

1. X-axis: time in ns.
1. Y-axis: entry/exit events per frame, per ns, or cumulative events depending on **Entry/Exit mode** and **Normalize** settings.
1. If **entries** and **exits** are missing, the GUI derives events from changes in **n_solvent** and shows a warning.

Histogram

1. X-axis: the selected per-frame metric (default: **n_solvent**, an integer count per frame).
1. Y-axis: frame count, or fraction of frames if **Normalize** is checked.
1. **Split zeros** shows a separate bar chart of zero vs non-zero frames and computes mean/median on the non-zero subset.
1. Red line: mean. Green line: median.
1. Interpretation limits: if most frames are zero occupancy, use the Timeline, Event Raster, or Matrix/Heatmap to see when occupancy occurs.

Matrix / Heatmap

Two modes:

1. **Solvent occupancy (top N)**: binary presence matrix for the top N solvents by occupancy.
1. X-axis: time in ns.
1. Y-axis: solvent rank (row 0 is the most occupied solvent).
1. Color: 1 = present, 0 = absent.
1. **Residue hydration**: heatmap of **hydration_freq** values when residue hydration is configured.

Note: the heatmap requires per-frame solvent IDs. If **analysis.store_ids=false** or **--no-ids** is used, this plot is not available.

Event Raster

1. X-axis: time in ns.
1. Y-axis: solvent rank (same ordering as the heatmap).
1. Points show occupancy events; **Segments** shows continuous occupancy segments and respects **Min duration**.
1. Interpretation limits: raster density depends on stride and top N; high stride can hide short events.

Frame extraction (GUI + CLI)

Rule format

Rules are simple comparisons against per-frame metrics. Examples:

1. **n_solvent>=1**
1. **entries>=2**
1. **occupancy_fraction==1**

occupancy_fraction is treated as a per-frame indicator (1 if **n_solvent>0**, else 0) when used in extraction rules.

CLI extraction example

```
sozlab-cli extract \
  --project /path/to/project.json \
  --soz shell_only \
  --rule "n_solvent>=2" \
  --min-run 10 \
  --gap 2 \
  --out /path/to/extracted
```

Outputs:

1. **<prefix>.xtc**: trajectory subset containing only selected frames.
1. **<prefix>_ref.pdb**: reference structure from the first selected frame.
1. **<prefix>_frames.csv**: manifest (frame index, time, n_solvent, optional solvent_ids hash).

1. **<prefix>_params.json**: extraction parameters (rule, min_run_length, gap_tolerance, counts).

How to validate extraction

1) Check **<prefix>_frames.csv** for the expected number of frames and time range. 2) Compare the manifest against **per_frame.csv** for consistency. 3) Load **<prefix>_ref.pdb** and **<prefix>.xtc** in a viewer to verify the selected event windows.

Reproducibility and provenance

1. Use a version-controlled project JSON and keep **metadata.json** from each run.
1. Record the SOZLab version (from **pyproject.toml**) and dependency versions (included in **metadata.json** QC summary and in the report).
1. Keep preprocessing notes in **inputs.processed_trajectory** and **inputs.preprocessing_notes**.
1. For deterministic comparisons, keep **analysis.frame_start**, **analysis.frame_stop**, **analysis.stride**, and **analysis.gap_tolerance** fixed.

Scientific QC / Reliability

Checklist

1. Preflight passes with no errors (**Project Doctor** or CLI run).
1. Solvent resnames match the topology (**water_resnames** and optional **ion_resnames**).
1. Seed selections resolve to the expected counts.
1. PBC box vectors are present (QC summary **pbc_summary.has_box** is true).
1. Time axis is monotonic and **dt** matches expected frame spacing.
1. **sozlab-cli validate --project ...** returns zero mismatches for sampled frames.

Minimal validation pathway

1) Run validation:

```
sozlab-cli validate --project /path/to/project.json --max-frames 200
```

2) Inspect **metadata.json** and the QC Summary panel for warnings. 3) Spot check **per_frame.csv** for plausible **n_solvent** ranges and time spacing.

Acceptance criteria (plots)

1. Timeline shows expected occupancy windows with reasonable noise for the chosen stride.
1. Histogram mean/median are consistent with the timeline (no contradictions).
1. Heatmap/raster show events when **n_solvent** is non-zero (if solvent IDs are stored).

Troubleshooting

Symptom: "Selection 'X' resolved to 0 atoms"

Most likely cause: selection string does not match the topology (wrong resname, resid/resnum, chain or segid). How to confirm: use the GUI Selection Tester or run a quick MDAnalysis **select_atoms** check. Remediation: adjust the selection string or use structured fields in the project file.

Symptom: "Water resnames did not match any residues"

Most likely cause: solvent resnames in the project file do not match the topology. How to confirm: print residue resnames from MDAnalysis. Remediation: update **solvent.water_resnames** (and **ion_resnames** if used).

Symptom: "No valid box vectors found; PBC distances may be unreliable"

Most likely cause: trajectory lacks box dimensions or they are zero. How to confirm: inspect **u.trajectory.ts.dimensions** in MDAnalysis. Remediation: regenerate or rewrap the trajectory with box info.

Symptom: Heatmap/Event Raster shows "No solvent IDs available"

Most likely cause: per-frame IDs are disabled (**analysis.store_ids=false** or **--no-ids**). How to confirm: check **per_frame.csv** for the **solvent_ids** column. Remediation: re-run with **store_ids=true**.

Symptom: Entry/Exit plot looks flat or zero

Most likely cause: per-frame entries/exits are zero or derived from **n_solvent** with a large stride. How to confirm: inspect **per_frame.csv** for **entries** and **exits** values. Remediation: re-run with smaller **analysis.stride** or ensure per-frame output is enabled.

If plots look wrong

1. Time axis mismatch: confirm **per_frame.csv** time units and the GUI conversion to ns.
1. Units mismatch: confirm cutoffs are in **A** or **nm** and match your expectations.
1. Seeds resolving incorrectly: check selection counts in Project Doctor and Selection Tester.
1. Solvent naming mismatch: check solvent resnames and oxygen names in the project file.